11/PART

SYSTEM AND METHOD FOR MANAGING COMPUTER APPLICATIONS SECURITY

INS A1

The invention concerns data processing systems and more particularly, in such systems, a system and method for managing the conditions for access to the different applications which are liable to be implemented by these data processing systems. The invention is preferentially but not limitatively intended to be implemented in the microprocessors of smart cards, whatever the field of use : health, banking, transport, mobile telephones etc.

The known security management methods have the following principal drawbacks :

- the first drawback is an obligation to have a hierarchy for selecting an application, that is to say it is necessary to pass through an imposed selection path commencing with the "grandparent" application and then the "parent" application in order to arrive at the "child" application, that is to say a selection path

similar to the one for selecting a file in a directory on a hard disk; in addition, no provision is made with regard to security.

There is therefore no relationship between the selection level and the security level.

- The second drawback is limiting the number of security levels or the number of applications. This is because a security register is dedicated to each application, the said register storing the rights acquired by this application through knowledge of secrets. In order to add n levels, that is to say to have a multiapplication series, it is necessary to associate, for example, a security register with each application, which results in using a large part of the high-speed memory where the security registers are stored. As the capacity of this high-speed memory is limited, it is not desirable to store many security registers therein. Thus, in certain systems, the number of hierarchical levels or the number of applications has been limited to three, that is to say three security registers.

- The third drawback is preventing the simple "emancipation" of the applications, that is to say making a "child" application independent of its "parent" application. This is because, when a new application is created, it is essential to use the rights and secrets of the "parent" application, which are the only ones available, until the secrets peculiar to the "child" application are created.

A

Summary of the Invention

The aim of the present invention is to implement a method of managing the security of data processing applications which does not have the drawbacks disclosed above and which therefore makes it possible :

- not to be limited with regard to the number of hierarchical levels or number of applications, and ;
- to make a "child" application independent of the "parent" application without passing through the latter from the security point of view.

The invention therefore concerns a system of managing the security of data processing applications, characterised in that :

- the data processing applications are recorded in directory files organised in an n-level tree, the level 1 directory being the highest level, and ;
- a number r of security registers which can each be allocated to a single directory and each security register containing all the rights or secrets S1 to Sp which have been granted under a directory.

The invention also concerns a method of managing the security of data processing applications in the management system described above, characterised in that it comprises the following steps consisting of :

(a) storing in security registers the rights granted under a directory according to given rules ;

(b) seeking in the tree the secrets presented, and

(c) verifying the knowledge equivalent to one or more rights at the level of the data processing application in order to satisfy the access conditions.

Brief Description of the Drawing

Other characteristics and advantages of the present invention will emerge from a reading of the following description of particular example embodiments, the said description being given in relation to the accompanying drawings, in which :

- Figure 1 is an example of a directory tree structure ;

- Figures 2.1 to 2.14 illustrate examples of the application of three rules for the allocation or deallocation of a security register to or from a directory ;

- Figures 3.1a to 3.6a and 3.1b to 3.6b illustrate examples of the application of the rule for the presentation of a secret ; and

- Figures 4.1 to 4.6 illustrate examples of the application of the rule for verifying the granting of the required right.

Detailed Description

The invention will be described in its application to a smart card and, more precisely, to a microprocessor used in a smart card. However, it is also applicable to any data processing system where it is necessary or simply desirable for certain services or functions offered by the system to be accessible only to certain users or operators.

In the case of smart cards, for example a bank card or a mobile telephone card, the services or functions which are available to the user may be subject to authorisation according to the type of subscription taken out, these authorisations (or rights) being granted by proving the knowledge of secrets which allow

access to the files necessary for the use of the service or function.

In the remainder of the description, the following definitions will be adopted :

- A file is a set of data able to be protected by access conditions ;

- A directory Rep is a set of files and/or directories in accordance with an arborescent organisation (Figure 1); normally a directory is dedicated solely to one application ;

- The conditions for access to a file or directory Rep define the criteria to be fulfilled, such as the presentation of a secret code or an external authentication, in order to be able to effect such or such a function on the file or directory ;

- The files and directories are organised in a tree with several levels, where the highest-level directory (level 1) is referred to as the "root" directory, or root of the tree. A level characterises the directories having the same hierarchical degree. The use of directories makes it possible to structure the data in a smart card. In Figure 1, only the directories Rep1, Rep2, Rep31, Rep32, Rep41, Rep42, Rep51 and Rep52 have been presented and each can contain one or more files. The directory Rep1 is the root of the tree comprising n=5 levels of directories, the directories Rep41 and Rep42 belonging to the level i=4 ;

- A security register R contains all the rights which have been granted under a directory and a right is the proof of the knowledge of a secret which is

identified by a reference such as a name, a number or an identifier. There are several ways of proving knowledge of a secret, for example by exchanging the value of the secret between the terminal and smart card or by exchanging data calculated by means of this secret: the operation is called presentation of the secret.

In general terms, the foundation of security on a smart card is to be able to make the use of the service or the function of the smart card dependent on proof of the knowledge of one or more secrets. Thus, in order to be able to use a function of the card, it is necessary :

- for the smart card to previously store this proof of the knowledge of the secret or secrets in a security register ;

- for the bearer of the smart card or terminal to prove that he has knowledge of the secret or secrets protecting the function ;

- for the card to verify, when the function is used, that the secret or secrets are indeed known.

The invention lies in the steps of the method consisting of :

(a) storing in the smart card the knowledge of the secret or secrets, that is to say the rights granted, according to the rules of allocation and deallocation of a security register to a directory ;

(b) seeking in the tree the secret or secrets presented ;

(c) verifying the knowledge of the secret or secrets in order to fulfil the access conditions.

To store the knowledge of a secret in a smart card (step (a)), it is necessary to correctly present the secret, which amounts to proving that the outside, for example a terminal or a card carrier, has knowledge of the said secret, this knowledge conferring on it or him the right to use functions or services offered by the card. It is the right which is stored in a security register at the rate of one register per application.

A security register R comprises a number p of digits or positions, each position being allocated to the knowledge of a secret corresponding to a granted right. A register with p=8 positions can record the knowledge of eight secrets S1 to S8, which will correspond to eight rights granted.

The number r of security registers R can be any number and the example which will be described will include r=3 of them. The security registers are not dedicated to a given level or directory as in the prior art and the link between a directory and a security register is dynamic, that is to say this link can be created or broken in accordance with the rules of the method according to the invention.

In order to store a right in a directory, it is necessary first of all to allocate or deallocate a security register to or from a directory in accordance with the following three rules RG1 to RG3 :

Rule RG1 :

A register is allocated to the current directory as soon as a right is granted under this directory, for example a secret code or an authentication. If a right

has already been granted under this directory, the register dedicated to it is updated.

Rule RG2 :

Selection of a new directory gives rise to the loss of the link connecting the old current directory to its security register except if the directory selected is the "child" of the old current directory.

Rule RG3 :

If the number r of security registers is saturated, that is to say the R=3 in the example described are used, the register which was allocated first, that is to say the highest level in the tree, is allocated to the new current directory in accordance with rule RG1.

It should be noted that the application of rule RG2 makes it possible to allocate two security registers to the same level, so that the allocation of a security register to a directory may be represented by a hierarchical level Ni allocated to the security register concerned, i varying from 1 to n.

Figures 2.1 to 2.14 illustrate applications of the rules RG1, RG2 and RG3. In these figures and the others, a black circle designates a directory, a grey circle designates a selected directory and a white circle designates a selected directory with a right removed.

Figure 2.1 illustrates the absence of selection of a directory whilst Figures 2.2 and 2.3 illustrate respectively the selection of the directories Rep1 and Rep2.

Thus the application of rule RG1 is illustrated in Figures 2.4, 2.6, 2.8, 2.10, 2.12 and 2.14. Figure 2.4 illustrates the presentation of a secret under the directory Rep2 at level N2. Figure 2.6 illustrates the presentation of a secret under the directory Rep31 at level N3. Figure 2.8 illustrates the presentation of a right under the directory Rep41 at level N4. Figure 2.10 illustrates the presentation of a right under the directory Rep51 at level N5. Figure 2.12 illustrates the presentation of a right under the directory Rep41. Figure 2.14 illustrates the presentation of a right under the directory Rep42.

The application of rule RG2 is illustrated by Figures 2.5, 2.7 and 2.9 with regard to the maintenance of the link between a security register and its directory when a new "child" directory thereof is selected.

Figures 2.5, 2.7 and 2.9 illustrate respectively the selection of the directory Rep31, Rep41 or Rep51.

The application of rule RG2 is illustrated by Figures 2.11 and 2.13 with regard to the breaking of the link between a security register and its directory. Thus Figure 2.11 illustrates the selection of the directory Rep41 whilst Figure 2.13 illustrates the selection of the directory Rep42.

The application of rule RG3 is illustrated by Figure 2.10, in which the register allocated the earliest R1 is allocated to the new selected directory Rep51.

Step (a) consisting of storing the rights attached to the knowledge of the secrets being performed, step (b) consisting of seeking in the tree the secret presented by the bearer of the smart card or by the terminal can be implemented.

A secret presented at the level of an application confers a right of use at the level of this same application. Thus the successful presentation of a secret within an application with the hierarchical level Ni updates the security register dedicated to this hierarchical level, in accordance with rule RG1, even if the secret presented is physically situated in a higher hierarchical level.

The rule for presentation of a secret is as follows :

Rule RG4 :

The presentation of a reference secret S amounts to verifying that the bearer of the smart card or terminal knows the value of the first reference secret S found by running through the hierarchical axis of the current application to the root directory.

The presentation of the reference secret S at the level of the current application situated at the hierarchical level Ni is effected by means of the following intermediate steps consisting of :

(b1) seeking a reference secret S in the current directory, that is to say at the level Ni, by means of a security management system, and verifying the existence of this secret within the application ;

(b2) if this secret exists, verifying that the presentation of the secret has succeeded, for example value for a secret code, cryptogram for a key, etc.

If the presentation has succeeded, the right associated with reference secret S is granted at the level of the current application at level Ni.

If the presentation has failed, the right associated with the reference secret S is not granted and the attempted presentation is terminated.

(b3) If the reference secret S does not exist within the current application at level Ni, seeking whether a secret with the same reference exists within the parent application at a level N(i-1) of the current application.

(b4) If the secret exists at the level of the parent application at level N(i-1), verifying that the presentation has succeeded.

If the presentation has succeeded, the right associated with the reference secret S is granted at the level of the current application at level Ni.

If the presentation has failed, the right associated with the reference secret S is not granted and the attempted presentation is terminated.

(b5) If the reference secret S does not exist within the parent application at level N(i-1), seeking the reference secret S at level N(i-2) along the hierarchical axis, and so on as long as the existence of a reference secret S has not been discovered.

(b6) If the reference secret S has not been found, the attempted presentation is terminated.

Several examples of the application of rule RG4 are illustrated in Figures 3.1a to 3.6a and 3.1b to 3.6b. Figures 3.1a and 3.1b, 3.2a and 3.2b, 3.3a and 3.3b correspond to examples where the right is granted whilst Figures 3.4a and 3.4b, 3.5a and 3.5b, 3.6a and 3.6b correspond to examples where the right is not granted.

In Figure 3.1a, the secret S3 exists locally under the directory Rep41 and no register is allocated to the directory Rep41. In Figure 3.1b, knowledge of the secret S3 is proved; a register R3 is allocated to the directory Rep41 at level N4 and the right is granted.

In Figure 3.2a, the secret S3 exists locally under the directory Rep41 and a register R3 is already allocated to the directory Rep41. Knowledge of the secret S3 is therefore proved and the security register R3 allocated to the directory Rep41 is updated (S3) so that the right is granted (Figure 3.2b).

In Figure 3.3a, the secret S2 does not exist locally under the directory Rep41; a register R3 is already allocated to the directory Rep41 and a secret S2 exists at the same time under the directories Rep2, Rep1, Rep42 and Rep51. Knowledge of the secret S2 is therefore proved and the security register allocated to the directory Rep41 is updated so that the right is granted (Figure 3.3b).

In Figure 3.4a, the secret S2 does not exist locally under the directory Rep41; a register R3 is already allocated to the directory Rep41 and a secret S2 exists both under the directories Rep2, Rep1, Rep42 and

Rep51. Knowledge of the secret S2 is therefore not proved so that the security register R3 allocated to the directory Rep41 is not updated and the right is not granted (Figure 3.4b).

In Figure 3.5a, the secret S2 does not exist locally under the directory Rep41; a register R3 is already allocated to the directory Rep41 and a secret S2 exists at the same time under the directories Rep2, Rep1, Rep42 and Rep51. Knowledge of the secret S2 is therefore not approved so that the security register R3 allocated to the directory Rep41 is not updated and the right is not granted (Figure 3.5b).

In Figure 3.6a, the secret S2 does not exist locally under the directory Rep41; a register R3 is already allocated to the directory Rep41 and a secret S2 exists at the same time under the directories Rep2, Rep1, Rep42 and Rep51. Knowledge of the secret S2 is not proved so that the security register R3 allocated to the directory Rep41 is not updated and the right is not granted (Figure 3.6b).

Step (c) consists of verifying that knowledge of the secret or secrets for fulfilling the access conditions, that is to say verifying that the secret protecting use of a function and service of the smart card, is indeed known to the outside world, that is to say that the right required has indeed been granted.

To this end, the invention implements a fifth rule RG5 which is stated as follows:

Rule RG5 :

A function requiring knowledge of a secret S is authorised if and only if, running through the tree along the hierarchical axis of the current application towards the root application, the first secret S encountered is known, that is to say correctly presented, by at least one of the applications belonging to the tree section having the current application and the application containing the secret S as its delimiters, these applications being able to be merged if the secret S exists in the current application.

In order to perform step (c), the management system must perform the following steps consisting of :

(c1) verifying that a security register is associated with the current application at level Ni ;

(c2) authorising the function if the security register contains the required right and terminating the verification ;

(c3) seeking the existence of the reference secret S within the current application at level Ni if no security register is associated with the current application or if the associated register does not contain the required right ;

(c4) refusing the function and terminating the verification if the secret exists within the current application ;

(c5) verifying that a security register is associated with the parent application at level N(i-1) of the current application if the reference secret S does not exist within the current application at level Ni ;

(c6) authorising the function and terminating the verification if the security register associated with the parent application contains the right required for using the function ;

(c7) seeking the existence of the reference secret S within the parent application at level N(i-1) of the current application if no security register is associated with the parent application or if the associated security register does not contain the required right ;

(c8) refusing the function and terminating the verification if the reference secret S exists within the parent application at level N(i-1) ;

(c9) verifying that a security register is associated with the grandparent application at level N(i-2) of the current application along the hierarchical axis of the current application towards the root application, if the reference secret S does not exist within the parent application at level N(i-1),

and so on as long as the existence of the reference secret S has not been discovered ;

(c10) refusing the function and terminating the verification if the secret has not been discovered.

Figures 4.1 and 4.2 illustrate two examples of an authorised function whilst Figures 4.3, 4.4, 4.5 and 4.6 illustrate four examples of a refused function.

In Figure 4.1, the function is accepted since the secret S3 exists locally, and is known under the directory Rep41.

In Figure 4.2, the function is accepted since the secret S1 does not exist locally but is known under the directory Rep2.

In Figure 4.3, the function is rejected since the secret S3 exists locally under the directory Rep41 and no right has been granted under this directory.

In Figure 4.4, the function is rejected since the secret S3 exists locally under the directory Rep41 and, although a security register R3 is allocated to the directory Rep41, knowledge of the secret S3 has not been proved.

In Figure 4.5, the function is rejected since the secret S2, which does not exist locally under the directory Rep41, nor in the directory Rep31, exists under the directory Rep2 and no security register is allocated to the directory Rep2. It should be noted that the function is rejected although a secret S2 is known under the directory Rep1.

In Figure 4.6, the function is rejected since the secret S1 has not been found by running along the hierarchical axis from the directory Rep41 towards the directory Rep1, although a secret S1 exists under the directories Rep51 and Rep32.